



PRESENTATION DE PROJET D'APPLICATION

INSTAGRID

COMPÉTENCES ÉVALUÉE :

- *Interpréter les gestes sur écran tactile*
- *Mettre en place une architecture adaptée à son projet*
- *Créer un design responsive à partir d'un mockup*

CAHIER DES CHARGES

- *Description générale*.....*p1*
- *Design*.....*p2*
- *Fonctionnalités*.....*p3*
- *Contraintes techniques*.....*p4*

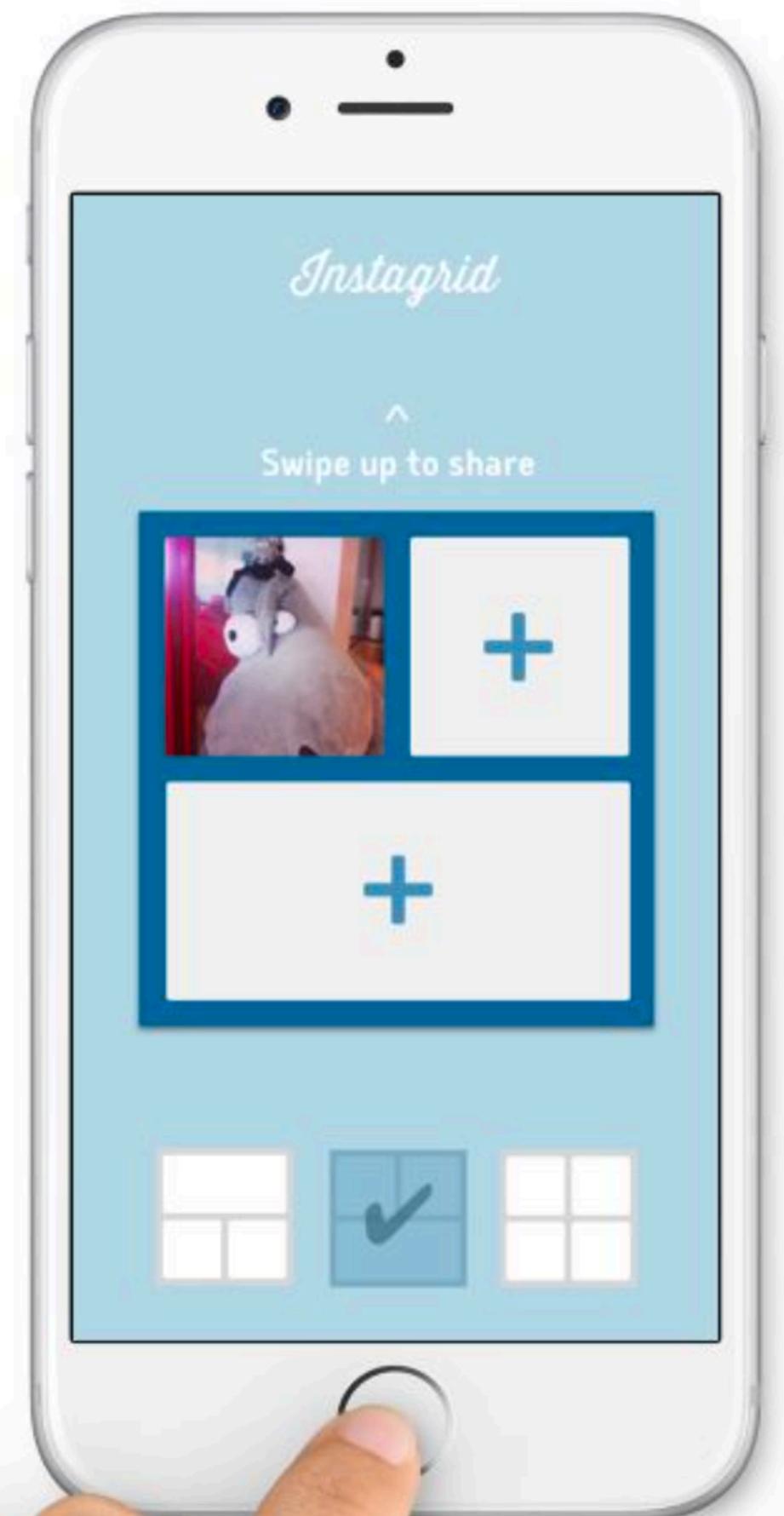
CREATION DU PROJET XCODE

- *Modele d'architecture*.....*p5*
- *Créer des interfaces graphique avec le storyboard*.....*p6*
- *Utiliser les méthodes dans le contrôleur*.....*p7*
- *Importer UIKit pour utiliser le UIView*.....*p13*

1. Description générale



Instagrid



Instagrid est une application iPhone permettant aux utilisateurs de combiner des photos en choisissant parmi plusieurs dispositions. Le résultat final est au format carré et partageable avec ses amis.

2. Design

The screenshot shows the Adobe XD interface with three screens displayed:

- iPhone Portrait:** Shows a light blue background with the word "Instagrid" at the top. Below it is a white box with a blue border containing a small image of a cartoon character and a large blue plus sign. A second, identical box is positioned below the first. At the bottom of the screen are three icons: a 2x2 grid, a 3x2 grid with a checkmark, and a 2x3 grid.
- iPhone Landscape:** Shows a light blue background with the word "Instagrid" at the top. It features two large white boxes with blue borders. The top box contains a small image of a cartoon character and a blue plus sign. The bottom box is empty and has a blue plus sign. To the right of these boxes are three smaller icons: a 2x2 grid, a 3x2 grid with a checkmark, and a 2x3 grid.
- Icon:** A blue square icon featuring a white camera shutter symbol.

The right side of the interface includes various toolbars and panels for design, including "Grille de répétition", "COMPOSANT", "MISE EN PAGE", and "APPARENCE".

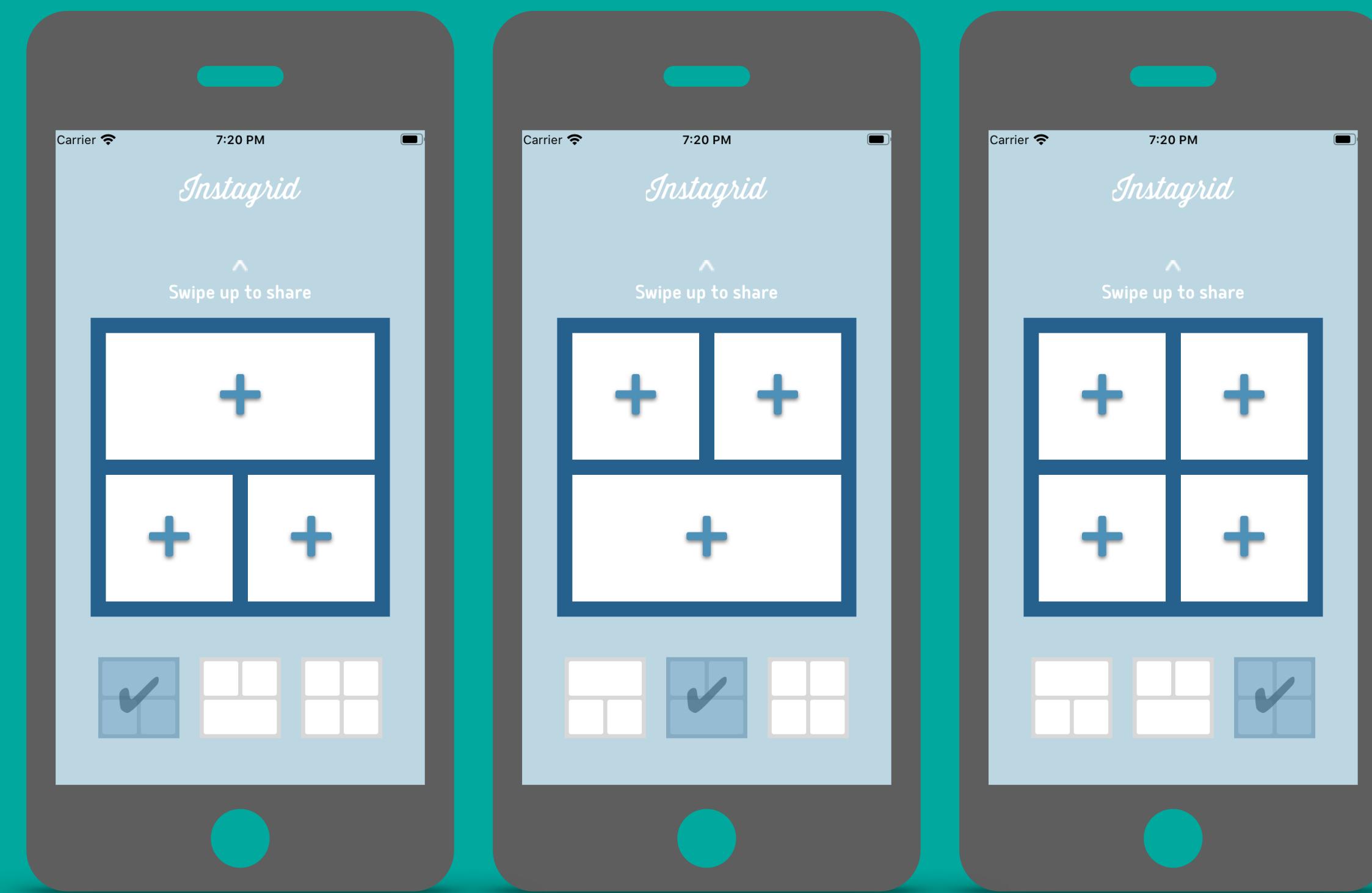
- Le design doit respecter celui du fichier Adobe XD fourni avec le cahier des charges
- Il doit être capable de s'adapter aux différentes tailles d'iPhone
- Il doit également être capable de s'adapter au mode portrait et paysage

3. Fonctionnalités

1 Sélection de la disposition des photos

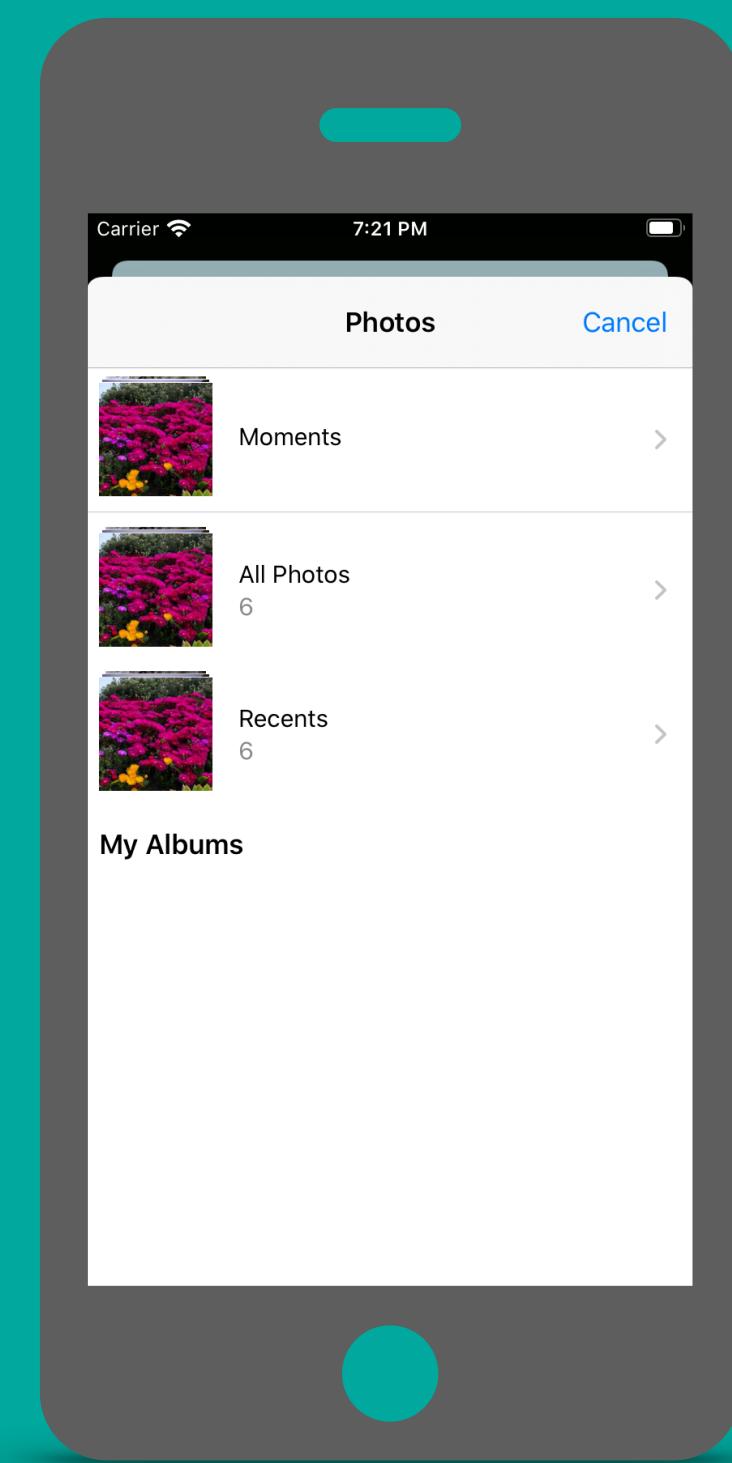
Les photos sont organisés selon une disposition que l'utilisateur peut choisir. Les trois dispositions disponibles sont rappelées (voir ci-dessous).

En tapant sur un bouton plus, l'utilisateur a accès à sa photothèque et peut choisir une des photos de son téléphone. Une fois choisie, celle-ci vient prendre la place de la case correspondante au bouton plus tapé.



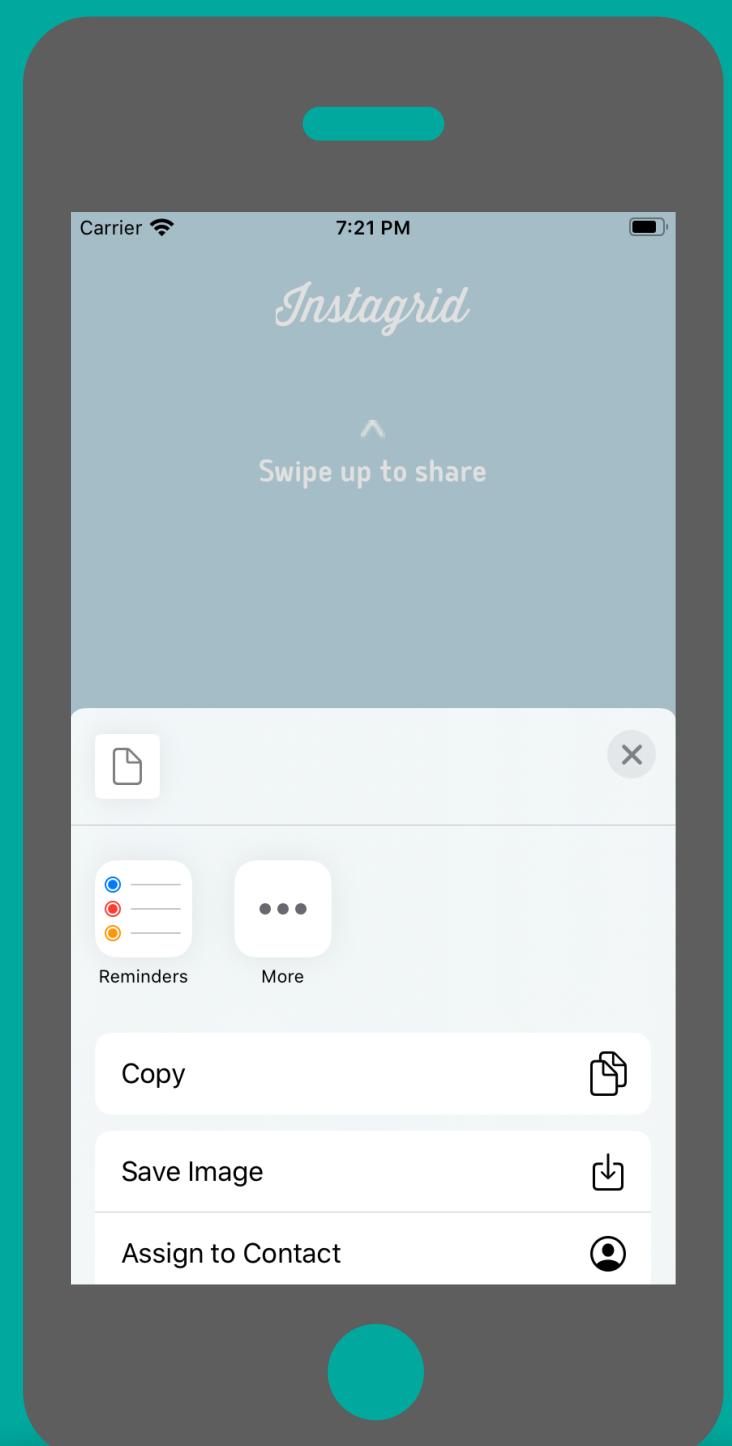
2 Photothèque

L'utilisateur a la possibilité de mettre ses photos dans la grille en allant les charger dans sa photothèque.



3 Swipe to share

Une fois l'animation terminée, la vue **UIActivityController** s'affiche et permet à l'utilisateur de choisir son application préférée pour partager sa création.



4. Contraintes techniques

- le langage utilisé doit être Swift 4 ou supérieur
- L'application doit être disponible à partir d'iOS 11.0
- L'application est supportée par toutes les tailles d'iPhone (de l'iPhone SE à l'iPhone XS Max)
- L'application n'a pas à être disponible sur iPad
- L'application supporte l'orientation Portrait et Paysage

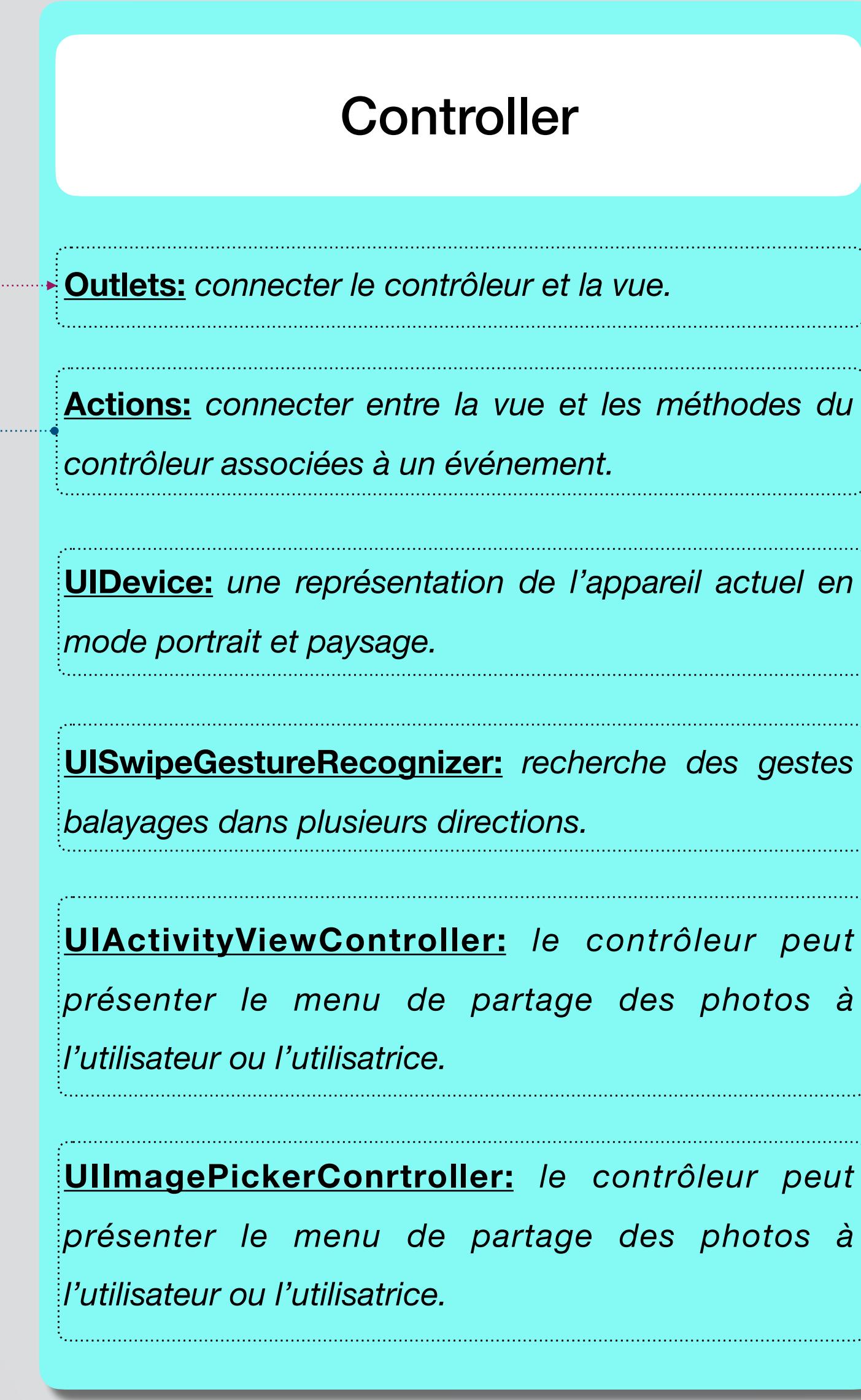
Disponible



Indisponible



Modele d'architecture: l'application se base sur le motif d'architecture MVC



Main.storyboard

Créer des interfaces graphique avec le storyboard

The screenshot shows the Xcode interface with the following details:

- Top Bar:** INSTAGRID > iPhone 8
- Central View:** Storyboard Editor showing the Main.storyboard scene for the View Controller Scene.
- Left Sidebar:** Project Navigator showing the project structure: INSTAGRID (INSTAGRID, MODEL, VIEW, Main.storyboard, LaunchScr..., CONTROLLER, SUPPORT, Products).
- Bottom Navigation:** View as: iPhone 8 (wC hR), zoom controls (58%), and orientation buttons (Device, Interface Style, Orientation, Vary for Traits).

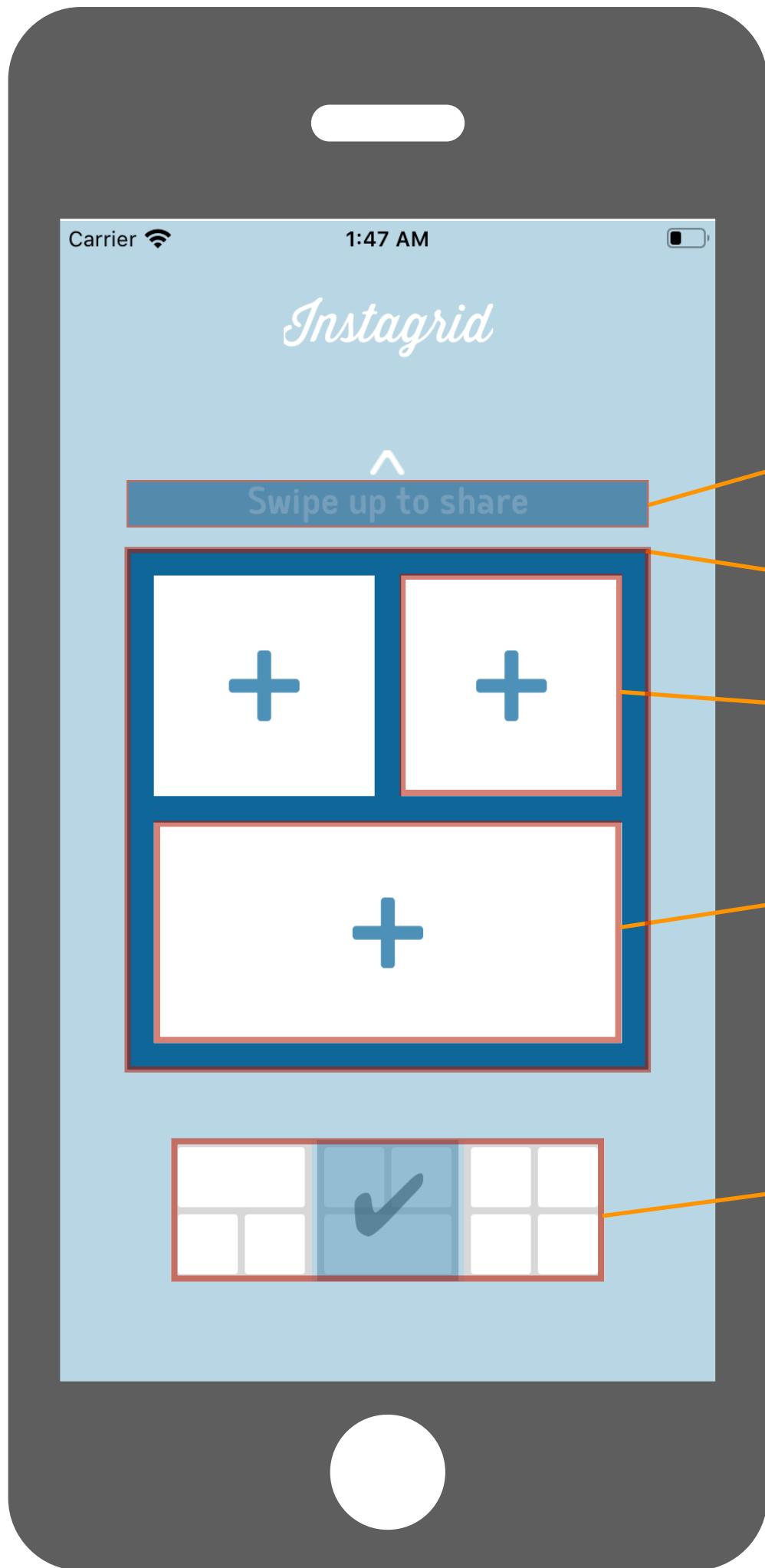
Annotations on the right side:

- Top Annotation:** "Le menu Ajouter de nouvelles contraintes se trouve en bas à droite de l'éditeur. On utilise ce menu pour ajouter de nouvelles contraintes aux éléments de l'interface utilisateur tels que les boutons et les étiquettes."
- Middle Annotation:** "Le menu Aligner se trouve à gauche du menu Ajouter de nouvelles contraintes . Ce menu crée des contraintes d'alignement. On utilise ce menu pour aligner verticalement une étiquette par rapport à une autre vue."
- Bottom Annotation:** "Le plan du document est sur le côté gauche. On utilise ce panneau pour afficher la hiérarchie de la disposition d'une vue. Ceci est utile pour déterminer la relation d'une vue."
- Bottom Right Annotation:** "L' inspecteur des attributs se trouve dans le volet utilitaire du milieu sur le côté droit de Xcode. On utilise ce panneau pour personnaliser les attributs d'une vue, par exemple lorsque vous changez la couleur d'arrière-plan d'une vue ou le texte d'un bouton."

CONTROLLER

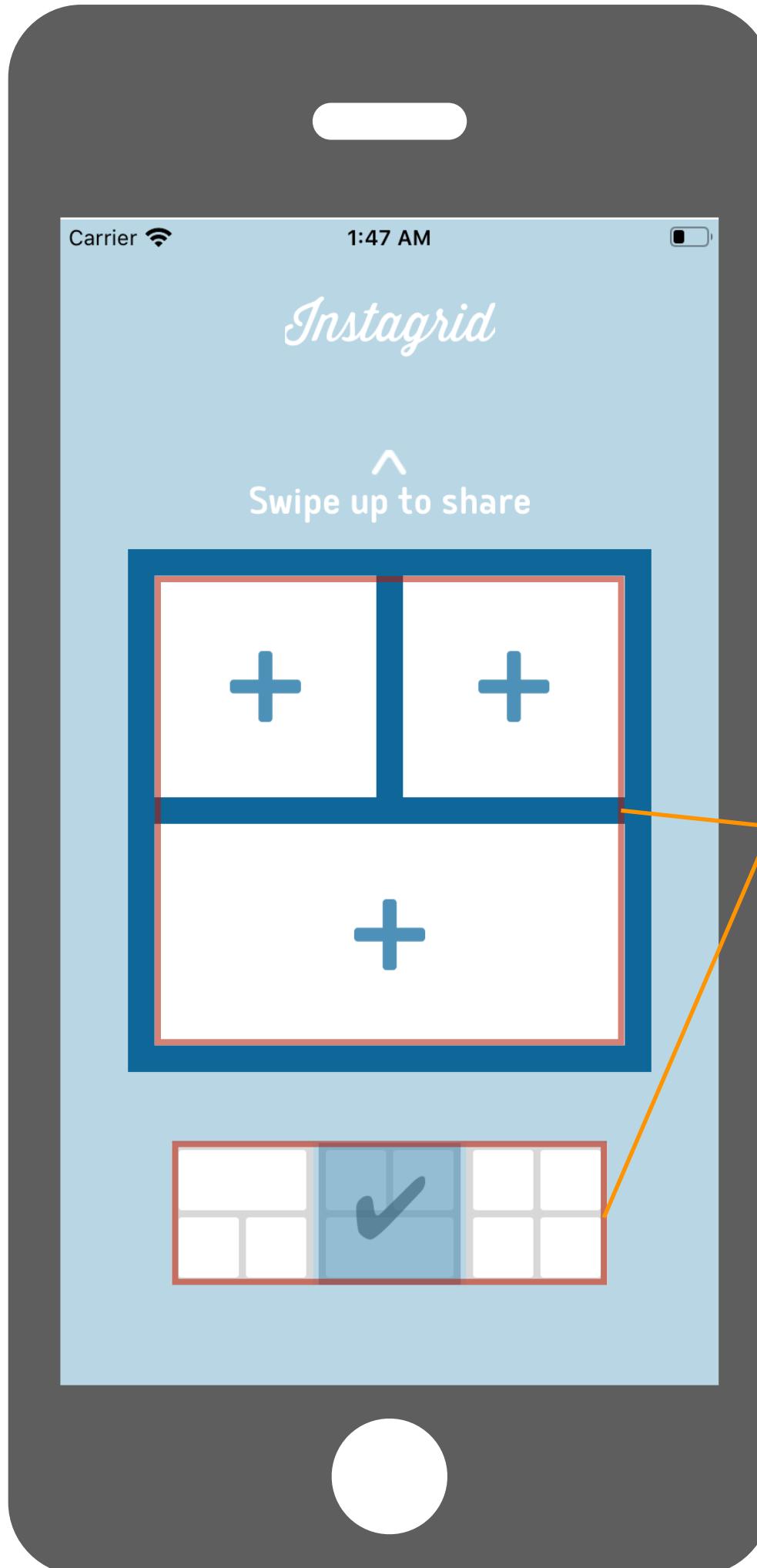
ViewController.swift

Outlets: connecter le contrôleur et la vue



```
//swipe  
@IBOutlet weak var swipeLabel: UILabel!  
  
//main grid  
@IBOutlet weak var mainGridView: UIView!  
@IBOutlet weak var topRightView: UIView!  
@IBOutlet weak var bottomRightView: UIView!  
  
//buttons  
@IBOutlet var groupButton: [UIButton]!
```

Actions: connecter entre la vue et les méthodes du contrôleur



```
// Utilisateur choisit le bouton
@IBAction func chooseButton(_ sender: UIButton) {
    groupButton.forEach { $0.isSelected = false }
    sender.isSelected = true

    switch sender.tag {
    case 1:
        topRightView.isHidden = true
        bottomRightView.isHidden = false
    case 2:
        topRightView.isHidden = false
        bottomRightView.isHidden = true
    case 3:
        topRightView.isHidden = false
        bottomRightView.isHidden = false
    default:
        break
    }
}

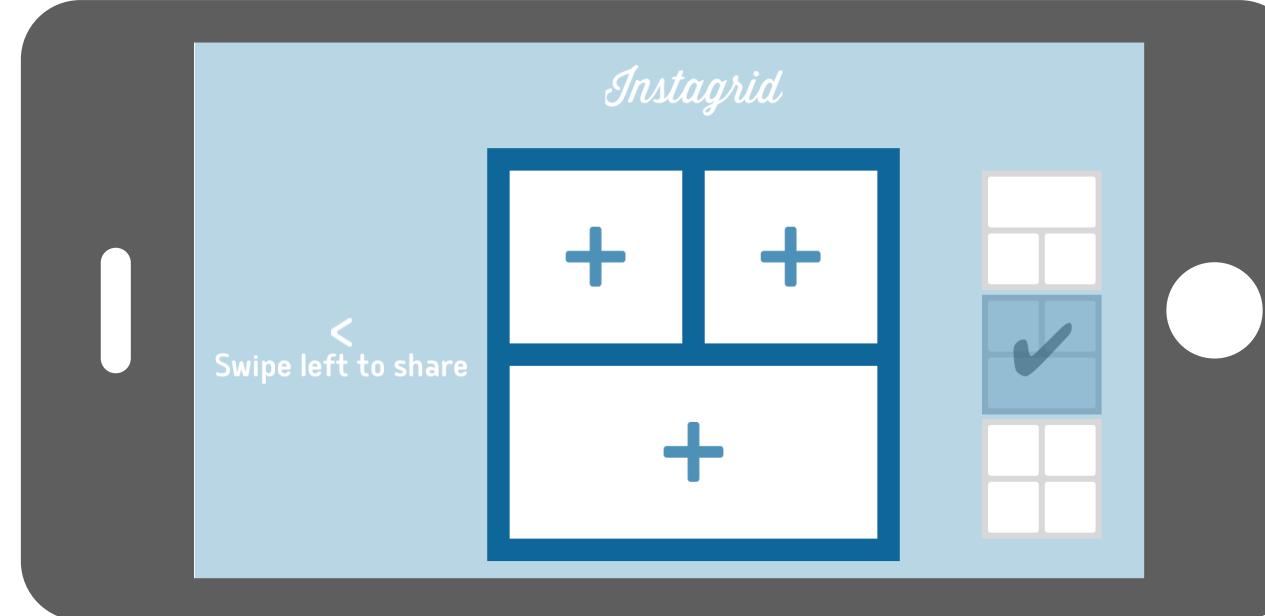
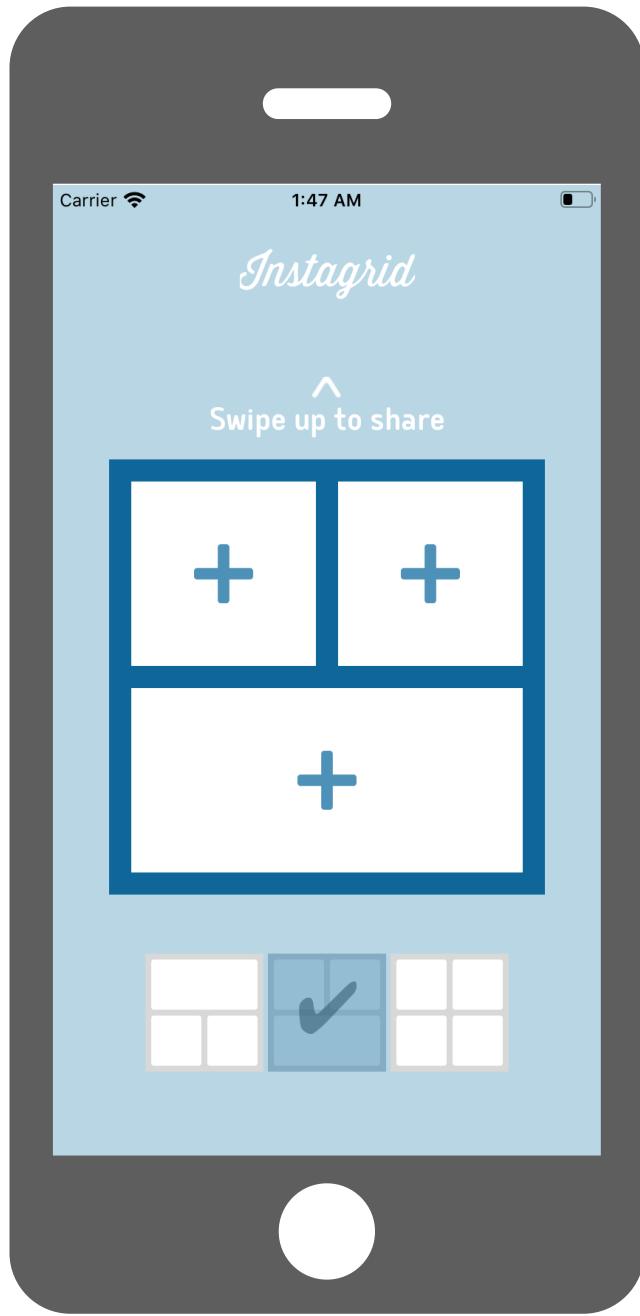
// Cette fonction permet d'importer les images
@IBAction func pickerUpImage(_ sender: UIButton) {
    selectButton = sender

    let actionSheet = UIAlertController(title: "Upload Photo", message: "Choose a source", preferredStyle: .actionSheet)

    if UIImagePickerController.isSourceTypeAvailable(.camera) {
        actionSheet.addAction(UIAlertAction(title: "Camera", style: .default, handler: { (action: UIAlertAction) in
            self.imagePickerController.sourceType = .camera
            self.present(self.imagePickerController, animated: true, completion: nil)
        }))
    }
    actionSheet.addAction(UIAlertAction(title: "Photo library", style: .default, handler: { (action: UIAlertAction) in
        self.imagePickerController.sourceType = .photoLibrary
        self.present(self.imagePickerController, animated: true, completion: nil)
    }))
}

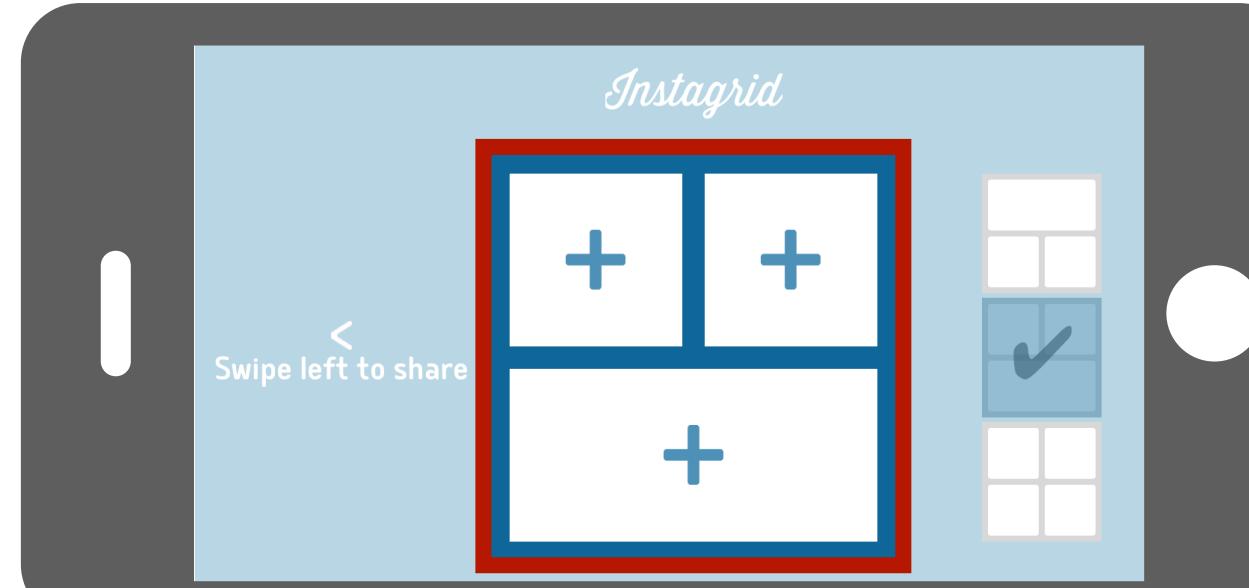
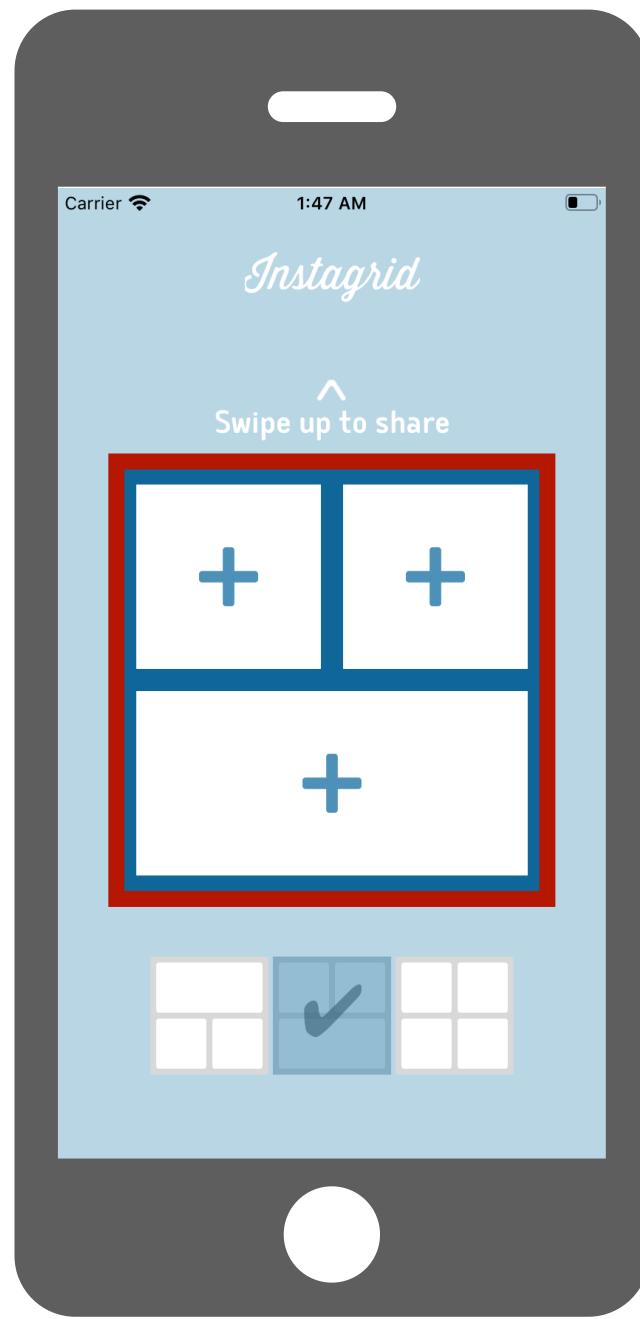
actionSheet.addAction(UIAlertAction(title: "Cancel", style: .cancel, handler: nil))
present(actionSheet, animated: true, completion: nil)
}
```

UIDevice: une représentation de l'appareil actuel en mode portrait et paysage



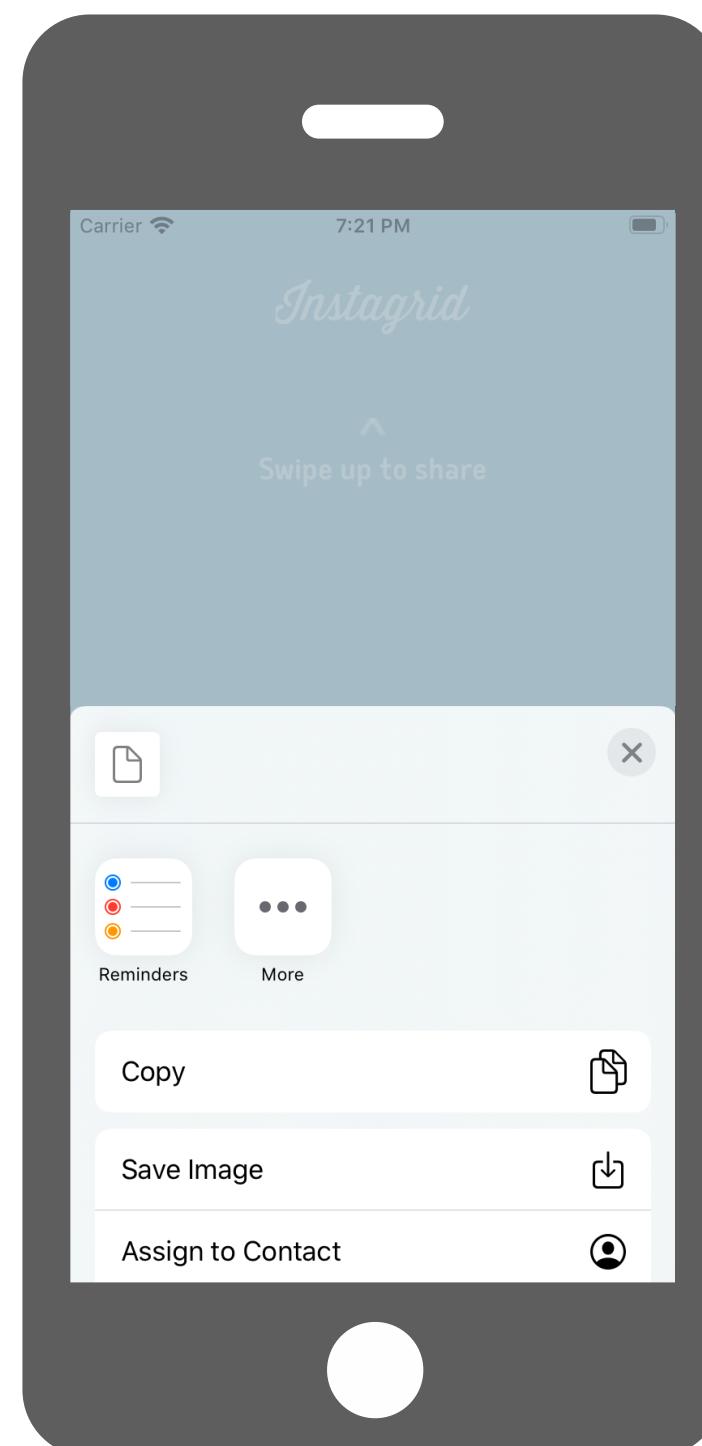
```
@objc private func settingDirection() {  
  
    if UIDevice.current.orientation == .portrait {  
  
        swipeGesture?.direction = .up  
  
        swipeLabel.text = "Swipe up to share"  
  
    } else if UIDevice.current.orientation == .landscapeLeft || UIDevice.current.orientation == .landscapeRight {  
  
        swipeGesture?.direction = .left  
  
        swipeLabel.text = "Swipe left to share"  
  
    }  
}
```

UISwipeGestureRecognizer: recherche des gestes balayages dans plusieurs directions



```
@objc private func swipeDirection(sender: UISwipeGestureRecognizer) {  
  
    if sender.direction == .up {  
        swipeUpToShare()  
    } else {  
        swipeLeftToShare()  
    }  
  
}  
  
// Animation: glisser vers le haut pour le partage  
private func swipeUpToShare() {  
    UIView.animate(withDuration: 1, animations: {  
        self.mainGridView.transform = CGAffineTransform(translationX: 0, y: -self.view.frame.height)  
    }) { (_) in  
        self.shareUIActivityViewController()  
    }  
}  
  
}  
  
// Animation: glisser vers la gauche pour le partage  
private func swipeLeftToShare() {  
    UIView.animate(withDuration: 1, animations: {  
        self.mainGridView.transform = CGAffineTransform(translationX: -self.view.frame.width, y: 0)  
    }) { (_) in  
        self.shareUIActivityViewController()  
    }  
}
```

UIActivityViewController: le contrôleur peut présenter le menu de partage des photos à l'utilisateur ou l'utilisatrice



```
// Elle permet de partager l'image, on utilise avec UIActivityViewController

private func shareUIActivityController() {

    let items = [mainGridView.transformedImage] // => voir fichier UIView.swift

    let ac = UIActivityViewController(activityItems: items , applicationActivities: nil)

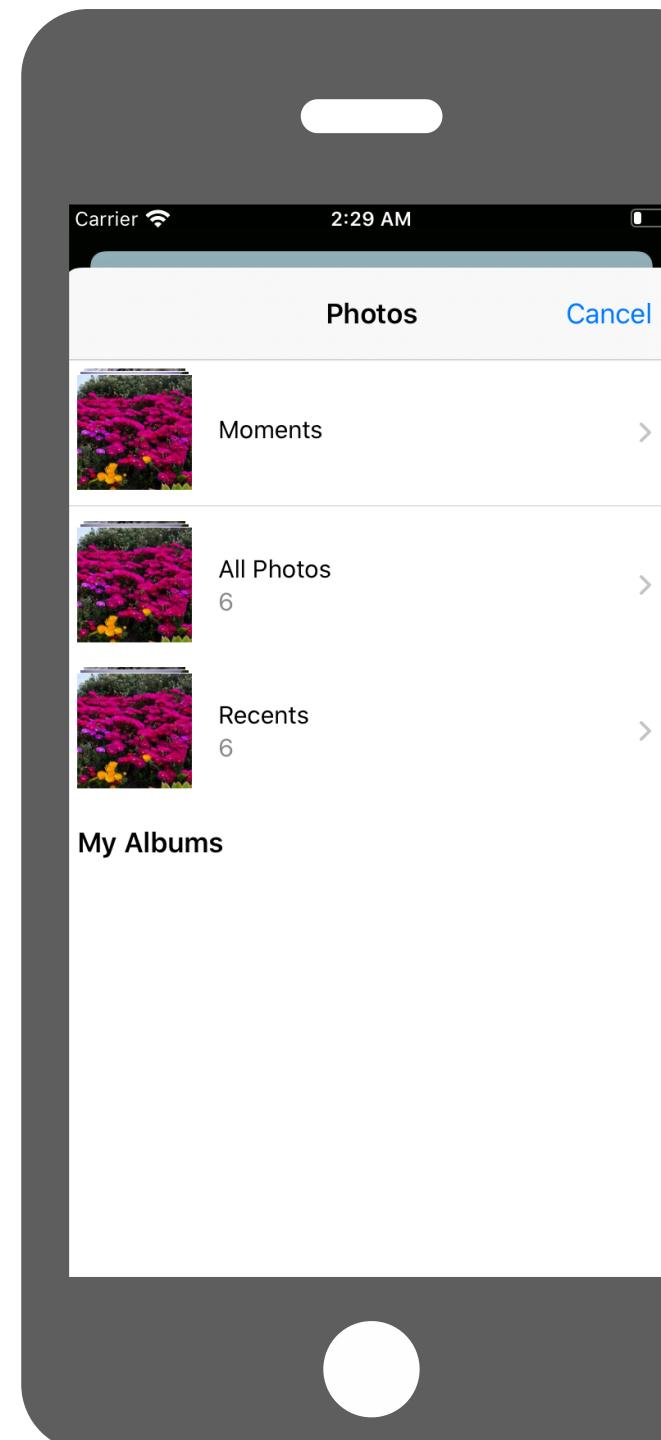
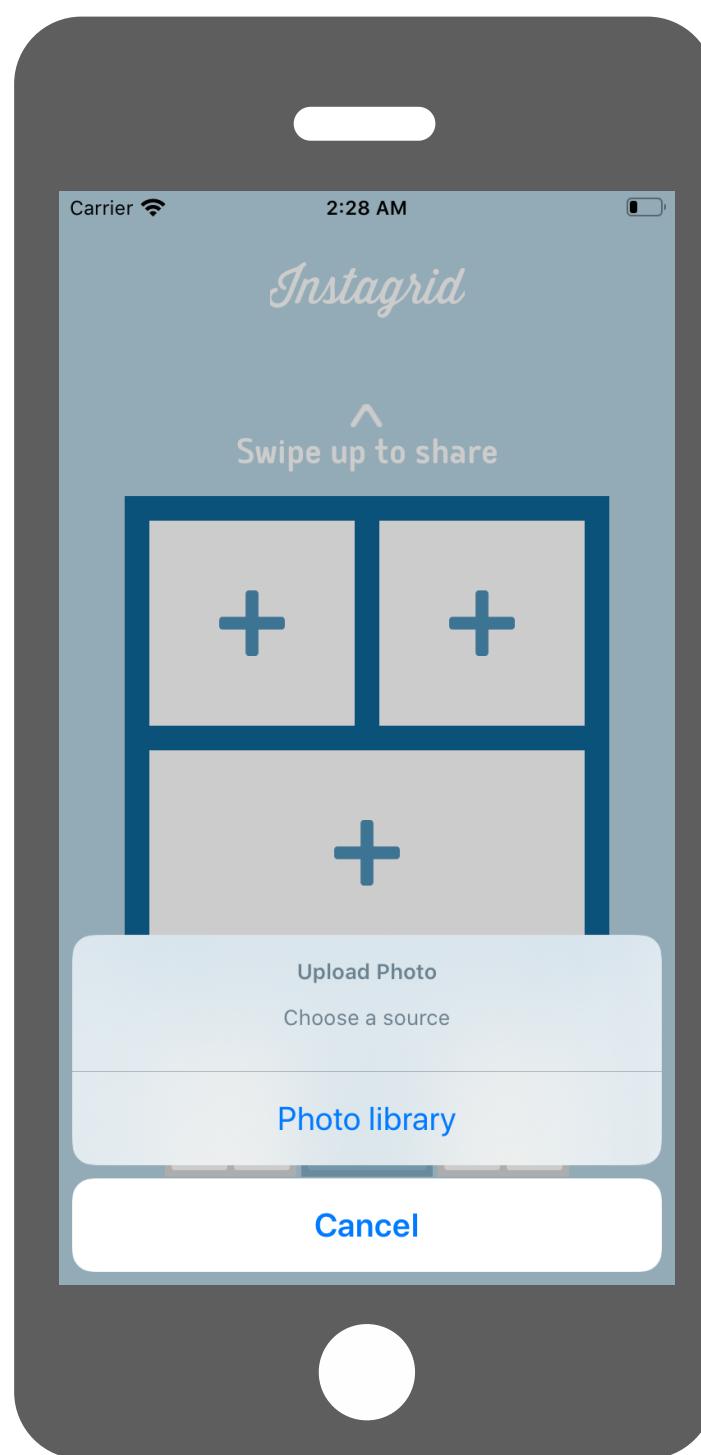
    present(ac, animated: true)

    ac.completionWithItemsHandler = { _, _, _, _ in

        UIView.animate(withDuration: 1) {

            self.mainGridView.transform = .identity // => Pour revenir la position d'origine = identity
        }
    }
}
```

UIImagePickerController: le contrôleur peut présenter le menu de partage des photos à l'utilisateur ou l'utilisatrice



```
// Func permet de mettre les images.
@IBAction func pickerUpImage(_ sender: UIButton) {
    selectButton = sender

    let actionSheet = UIAlertController(title: "Upload Photo", message: "Choose a source", preferredStyle: .actionSheet)

    if UIImagePickerController.isSourceTypeAvailable(.camera) {
        actionSheet.addAction(UIAlertAction(title: "Camera", style: .default, handler: { (action: UIAlertAction) in
            self.imagePickerController.sourceType = .camera
            self.present(self.imagePickerController, animated: true, completion: nil)
        }))
    }
    actionSheet.addAction(UIAlertAction(title: "Photo library", style: .default, handler: { (action: UIAlertAction) in
        self.imagePickerController.sourceType = .photoLibrary
        self.present(self.imagePickerController, animated: true, completion: nil)
    }))

    actionSheet.addAction(UIAlertAction(title: "Cancel", style: .cancel, handler: nil))
    present(actionSheet, animated: true, completion: nil)
}

//MARK:- METHOD: UIImagePickerController

extension ViewController: UIImagePickerControllerDelegate, UINavigationControllerDelegate{

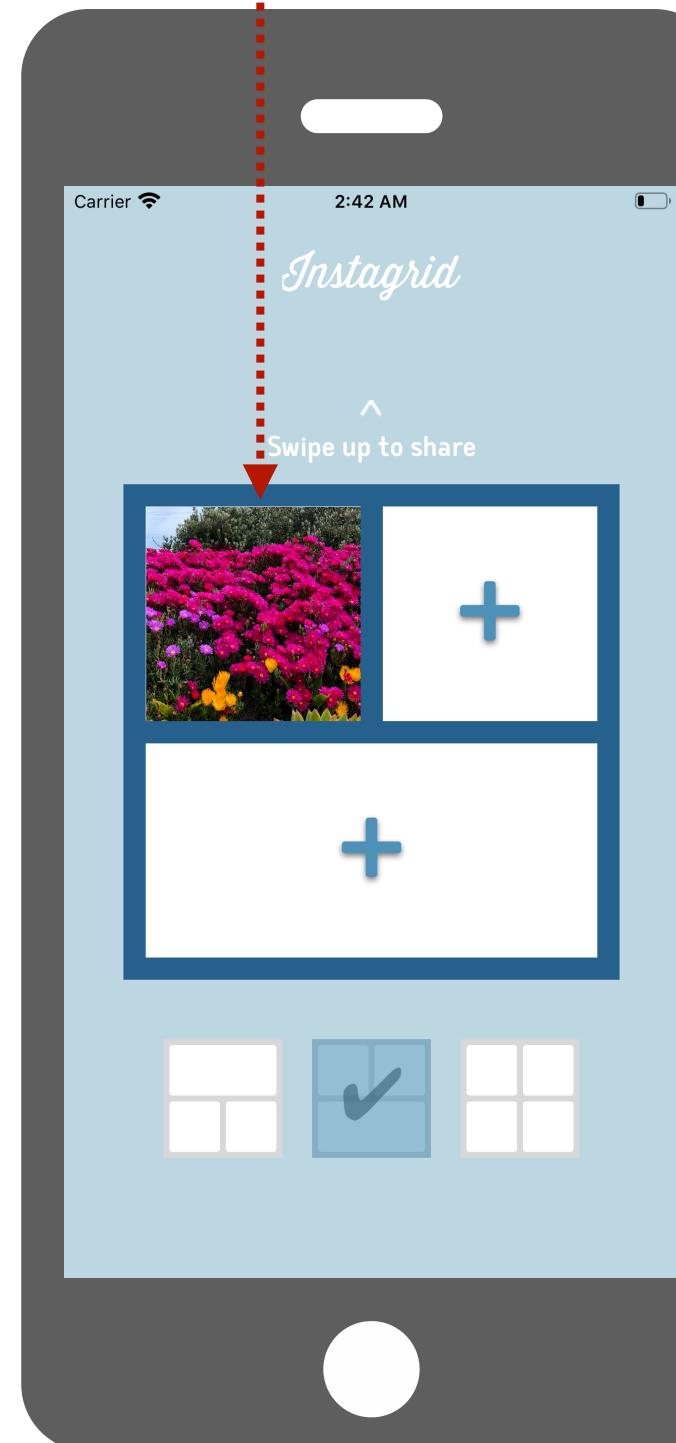
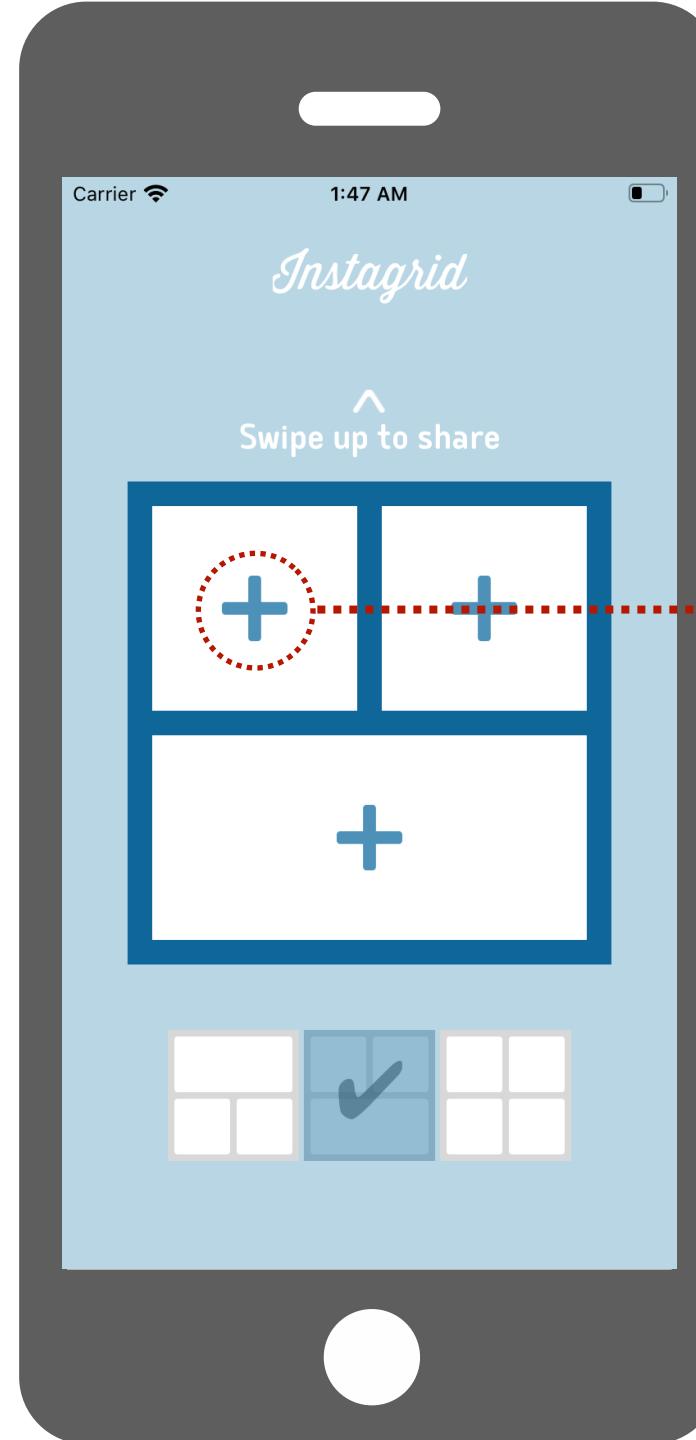
    func imagePickerController(_ picker: UIImagePickerController, didFinishPickingMediaWithInfo info: [UIImagePickerController.InfoKey : Any]) {
        // la variable image symbolise l'image que tu as sélectionné
        let image = info[.originalImage] as? UIImage
        selectButton?.setImage(image, for: .normal)
        selectButton?.subview.first?.contentMode = .scaleAspectFill

        picker.dismiss(animated: true, completion: nil)
    }
}
```

SUPPORT FILE

UIView.swift

UIGraphicsImageRenderer: permet de transformer UIView à un UIImage



```
import UIKit

extension UIView {

    // UIGraphicsImageRenderer: transforme en UIView à UIImage

    var transformImage: UIImage {

        let renderer = UIGraphicsImageRenderer(size: self.bounds.size)

        let img = renderer.image { ctx in

            self.drawHierarchy(in: self.bounds, afterScreenUpdates: true)
        }

        return img
    }
}
```

